Biblioteca de Interface Gráfica para Celulares

Orientador: Paulo César Rodacki Gomes
Aluno: Marcos Gorll

ROTEIRO

- Introdução
- Objetivos
- Fundamentação teórica e trabalhos correlatos
- Projeto e implementação
- Conclusão e resultados
- Extensões

INTRODUÇÃO

Mobilidade é uma tendência crescente, e está sendo cada vez mais explorada

Software para dispositivos móveis estão cada vez mais poderosos, e necessitam de melhores recursos

OBJETIVO GERAL

Criar uma API de interface gráfica para celulares, composta de vários componentes e com funcionamento semelhante ao funcionamento da API Sun AWT

OBJETIVOS ESPECÍFICOS

- Criar os componentes de interface
- Implementar sistema de eventos
- Implementar gerenciadores de *layout*
- Integrar com a API *Mobile* 3D *Graphics* (M3G)
- Criar aplicação de testes

FUNDAMENTAÇÃO

- Java 2 Micro Edition (J2ME) : Java para dispositivos móveis
- Configurações J2ME: Define a API que estará disponível para um grupo de dispositivos
- CLDC: Configuração disponível para celulares

FUNDAMENTAÇÃO

- *Profile*: Grupo de recursos específicos, que agregam funcionalidades as configurações
- MIDP: *Profile* disponível para celulares, com recursos de comunicação e componentes gráficos

FUNDAMENTAÇÃO

- M3G: Framework para desenvolvimento 3D em celulares
- AWT: API de interface gráfica para *desktop*
- JavaDoc: Documentação, em html, gerada a partir da documentação escrita no código fonte

TRABALHOS CORRELATOS

- *Java Tiny Gfx Library*: *Framework* para desenvolvimento de interfaces gráficas e construção de jogos
- Thinlet: API de interface gráfica, para uso em *pocket pcs* e *desktop*

Thinlet

- Baseado na API Sun AWT
- Telas são criadas em arquivos XML
- Motor lê os arquivos XML e cria as telas
- Possui sistema de *layout*
- Não possui sistema de *look and feel*

JTGL

- Não pode ser testado, *site* estava indisponível durante o desenvolvimento do trabalho

PROJETO E IMPLEMENTAÇÃO

- Requisitos Funcionais (RF)

1. Possuir os principais componentes de interface gráfica
2. Possuir sistema de eventos
3. Suporte a gerenciadores de *layout*
4. Suportar *look and feel*
5. Suportar sistemas de *driver* de teclado

PROJETO E IMPLEMENTAÇÃO

- Requisitos Não Funcionais (RNF)

1. Desenvolvido em JAVA
2. Baixo consumo de memória
3. Integração com M3G
4. Documentação em JavaDoc
5. Funcionamento semelhante ao da API Sun AWT

PROJETO E IMPLEMENTAÇÃO

- Ferramentas utilizadas na especificação

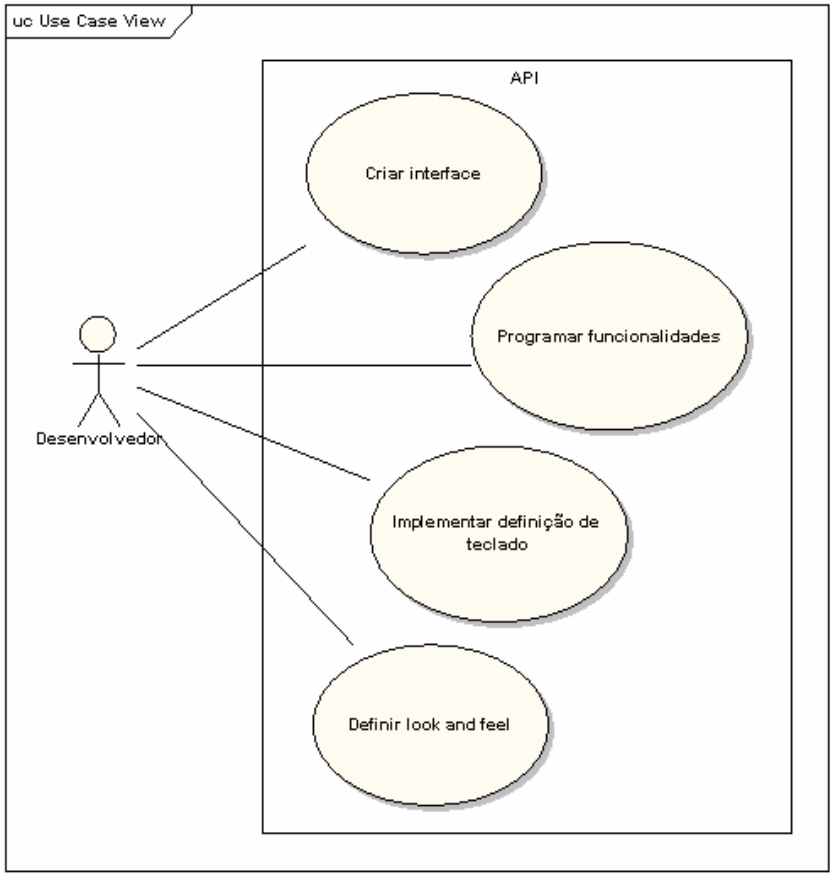
- Enterprise Architect

- Diagramas utilizados

- Caso de Uso, Classes e Seqüência

PROJETO E IMPLEMENTAÇÃO

- Casos de Uso



PROJETO E IMPLEMENTAÇÃO

- Criar Interface

É o caso de uso, onde o desenvolvedor modela e implementa a interface, não se preocupando com eventos e funcionalidades

PROJETO E IMPLEMENTAÇÃO

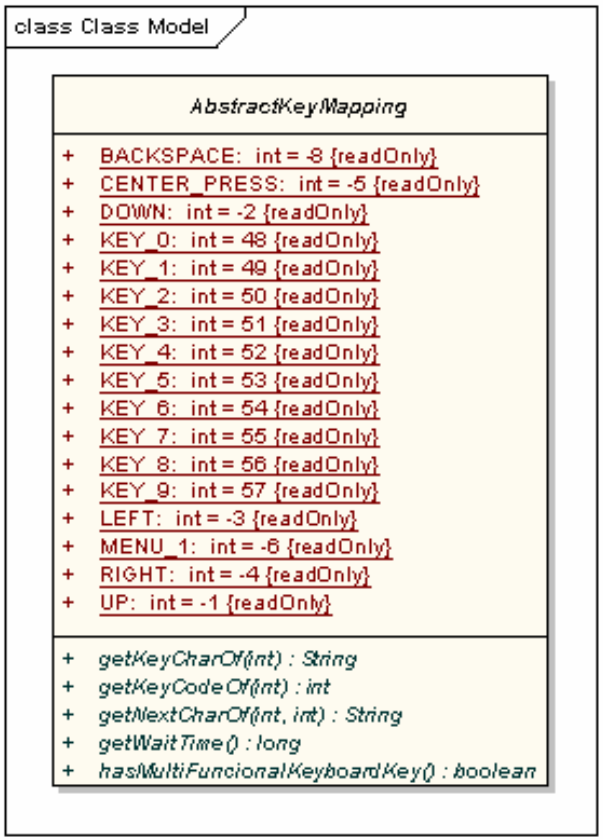
- Programar funcionalidades

É o caso de uso, onde o desenvolvedor implementa os seus eventos, ou seja, o comportamento da tela criada

PROJETO E IMPLEMENTAÇÃO

- Implementação de definição de teclado

É o caso de uso, onde o desenvolvedor cria o *driver* para o teclado do dispositivo alvo da aplicação



PROJETO E IMPLEMENTAÇÃO

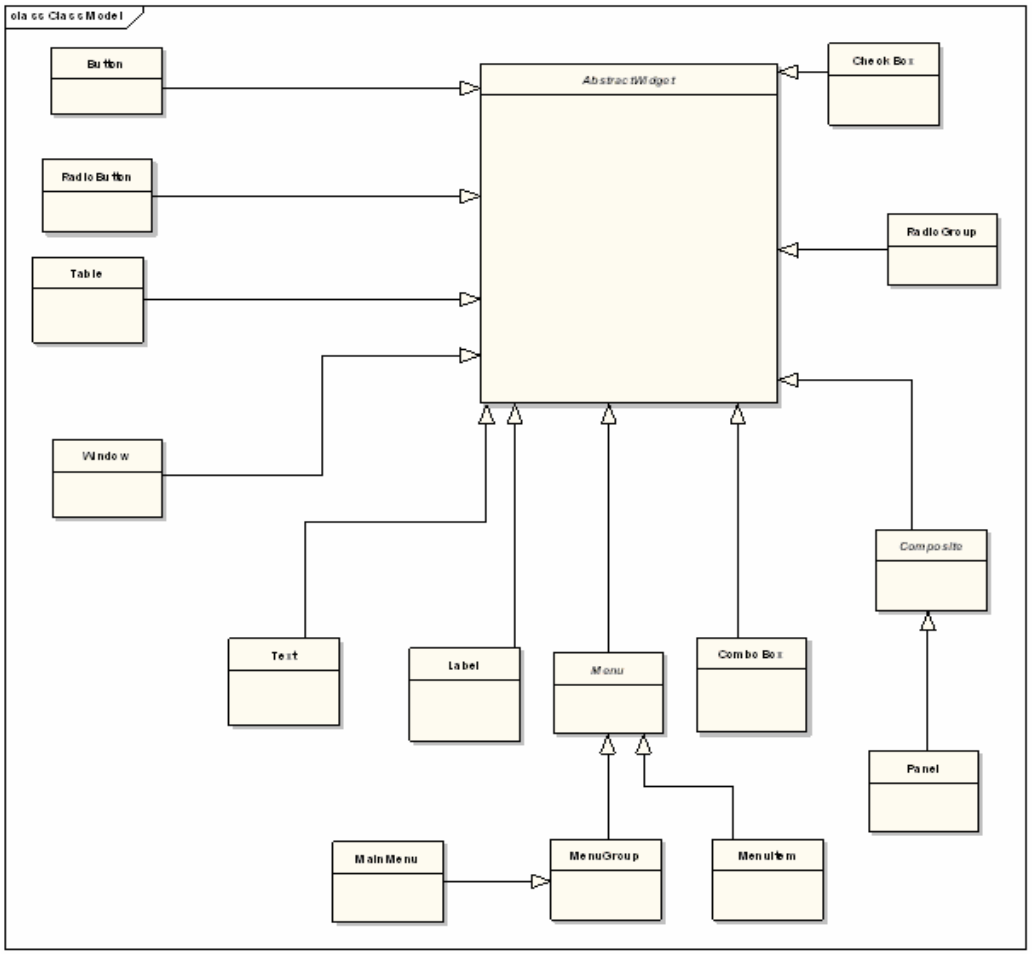
- Implementação do *look and feel*

É o caso de uso, onde o desenvolvedor implementa o seu prórpio *look and feel,* se desejar

PROJETO E IMPLEMENTAÇÃO

- Componentes gráficos

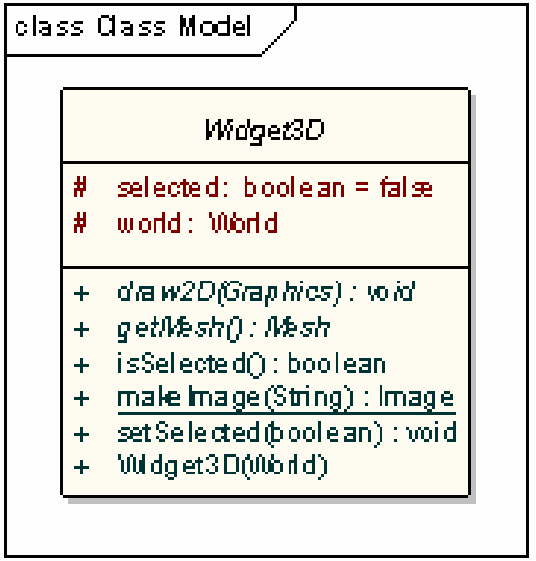
1. Button
2. Text
3. Label
4. Panel
5. ComboBox
6. CheckBox
7. Window
8. Table
9. RadioGroup e RadioButton
10. Menu



PROJETO E IMPLEMENTAÇÃO

- Integração com M3G

Para integração com a API M3G, foi criada uma especificação de componente 3D, que possui facilidade de tratamento de imagens e integração com as classes próprias da M3G



PROJETO E IMPLEMENTAÇÃO

- Ferramentas utilizadas

1. IDE Eclipse
2. Emulador de celular
3. Plugin para o eclipse, EclipseME

CONCLUSÃO E RESULTADOS

- A API está criada e funcionando
- Resultado sobre os requisitos:

1. Possuir os principais componentes de interface gráfica (OK)
2. Possuir sistema de eventos (OK)
3. Suporte a gerenciadores de *layout* (OK)
4. Suportar *look and feel* (OK)
5. Suportar sistemas de *driver* de teclado (OK)

CONCLUSÃO E RESULTADOS

1. Desenvolvido em JAVA (OK)
2. Baixo consumo de memória (OK)
3. Integração com M3G (+-)
4. Documentação em JavaDoc (OK)
5. Funcionamento semelhante ao da API Sun AWT (+-)

CONCLUSÃO E RESULTADOS

- Comparando-se a presente API, com Thinlet, pode-se constatar que:

1. Thinlet possui mais recursos
2. Thinlet possui mais componentes
3. Thinlet não possui sistema de *look and feel*
4. Thinlet funciona somente em *desktop* ou *pocket pcs* com suporte a configuração CDC

CONCLUSÃO E RESULTADOS

- Extensões e melhorias

1. Criação de um ambiente *drag and drop* para implementação das telas
2. Criação de novos componentes
3. Melhoria na integração com M3G